

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
15 November 2001 (15.11.2001)

PCT

(10) International Publication Number
WO 01/86570 A1

(51) International Patent Classification⁷: **G06F 17/60**

(21) International Application Number: PCT/US01/09909

(22) International Filing Date: 28 March 2001 (28.03.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/566,643 8 May 2000 (08.05.2000) US

(63) Related by continuation (CON) or continuation-in-part (CIP) to earlier application:
US 09/566,643 (CON)
Filed on 8 May 2000 (08.05.2000)

(71) Applicant (for all designated States except US): APOGEE NETWORKS, INC. [US/US]; Park 80 West - Plaza II, Saddle Brook, NJ 07663 (US).

(72) Inventors; and

(75) Inventors/Applicants (for US only): PRICE, John, D.

[US/US]; 5 Ashland Court, Colts Neck, NJ 07722 (US). FRENKIEL, Scott, Thomas [US/US]; 2 Woodlen Court, Freehold, NJ 07728 (US). SNOWDON, Arthur, Reed [US/US]; 39 Wyckham Road, Tinton Falls, NJ 07724 (US). JEFFERY, Donald, Charles [US/US]; 9 Kristen Court, Matawan, NJ 07747 (US). LEUNG, Yiu, Kau [US/US]; 5 Knob Hill Road, Morganville, NJ 07751 (US).

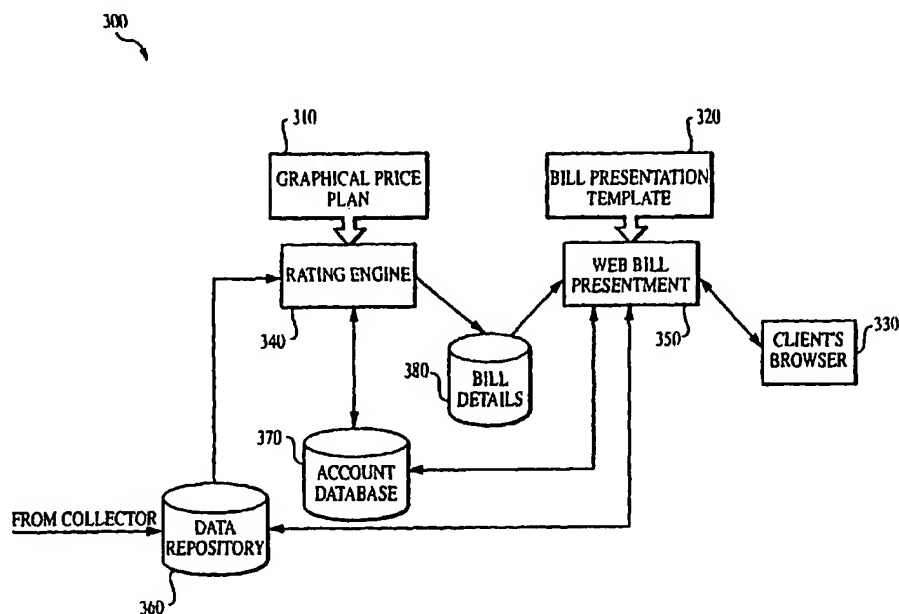
(74) Agent: BORODACH, Samuel; Fish & Richardson P.C., Suite 2800, 45 Rockefeller Plaza, New York, NY 10111 (US).

(81) Designated States (*national*): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) Designated States (*regional*): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European

[Continued on next page]

(54) Title: USAGE-BASED PRICE PLANNING



(57) Abstract: A method and apparatus (100, 200, 300) for usage-based price planning is disclosed. A Price plan (310) for services is established according to how the services are used (235). Charges are calculated according to the price plan (310) as applied to the actual usage (340). The apparatus (100, 200, 300) is used by network service providers (NSPs) (130, 230) and Internet service providers (ISPs) (130, 230) to determine billing for their customers (110, 210) based on how and to what extent the customers (110, 210) use the network connection provided by the NSP (130, 230) or ISP (130, 230).

WO 01/86570 A1



patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

Published:

— *with international search report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

USAGE-BASED PRICE PLANNING

This invention relates to networks and network usage, and more particularly to network pricing plans.

5

BACKGROUND

Service providers are in constant competition with each other to find new ways of attracting customers. An increasing number of service providers have entered the ever-growing world of e-commerce that demands high quality of service and quick turn-around time. E-commerce service providers offer a wide range of services that involve direct on-line services such as network access from Internet Service Providers (ISP) and Network Service Providers (NSP) and more traditional services such as e-commerce auctioning to further their service. For example, many on-line auction services use e-commerce to help sellers put their products up for auction while a number of buyers enter a bidding war leading to an ultimate purchase between the high bidder and the seller. All of these services found on-line involve some sort of usage, whether that usage be time spent on-line or bandwidth used during the time on-line. In particular, ISPs and NSPs provide network access to users who range from an individual web surfer who may be on-line a few hours at a time to large corporations who may be on-line 24 hours a day, seven days a week and who require a large amount of bandwidth.

Whether an individual surfer or a large corporation, customers attempt to find the best possible deals to minimize the cost of being on-line. ISPs and NSPs typically offer pricing plans to their customers that involve a charge per unit time, whether that be an hourly charge or an unlimited monthly charge. In addition, many ISPs and NSPs have an "acceptable use" condition for using their service. For example, customers cannot send "spam" emails over the link provided by the ISP or NSP. Another condition is that a customer cannot hook up a server to the link because the server can use an unacceptable amount of bandwidth. However, ISPs and NSPs typically are not equipped to monitor violations of these conditions. Furthermore,

increasing ISP competition drives the marketing departments of the ISPs to come up with quick changes in the ISP pay plans to meet the market demand.

SUMMARY

5 In general the inventions features a method and apparatus for representing pricing plans for on-line services, in particular for Internet and network services.

 In one aspect, the invention features a method of determining a billing-cycle price for network services, including establishing a price plan for each of a plurality of clients, wherein the price plan sets a price per use unit, collecting use data from the
10 plurality of clients, sorting the use data into data groups corresponding to each of the plurality of clients; and processing the data groups to determine the billing-cycle price for each of the plurality of clients, wherein the billing-cycle price is calculated from use data and the price per use unit.

 In an implementation, the price per use unit is determined by: time that each of
15 the clients used the network services, network bandwidth that each of the clients used, network content that each of the clients used, email that each of the clients generated on the network, electronic storage space used by each of the clients, web space that each of the clients used, multimedia usage that each of the clients used, or Internet protocol addresses in use by each of the clients.

20 In another implementation, the method further includes allowing each of the plurality of clients to choose a price plan that includes more than one price per use unit, calculating a billing-cycle price for each price per use unit, presenting each billing-cycle price to each of the clients, allowing the client to choose the billing-cycle price that is the most cost efficient, transmitting a bill presentation template to each of
25 the clients, wherein the bill presentation template is transmitted to a client web browser connected to the network, and displays charges presentation incurred during the billing cycle.

 In another aspect, the invention features a method of developing a price plan for services on a display, including providing an interface on the display, providing on

the interface images adapted to receive price plan criteria and receiving in the images price plan criteria.

In an implementation, the images on the interface are graphical user interface (GUI) screens.

- 5 In another implementation, providing the images comprises presenting the GUI screens in an order to present price plan criteria input options for generating the price plan to the user.

In another implementation, the method further provides generating the price plan into source code and compiling the computer code into machine-readable code.

- 10 In another implementation, the method provides supplying the price plan data to at least one of the source code and the machine-readable code to calculate a charge for the services.

- In still another implementation, the images on the interface are graphical elements representing components of the price plan and providing the images includes
15 arranging the graphical elements on the display that graphically represents the calculations for the price plan charge and providing interconnections among the graphical elements to provide price plan data flow paths.

- In yet another implementation, the method further includes storing the arrangement of graphical elements as source code, compiling the source code into
20 machine-readable code and supplying the price plan data to at least one of the source code and the machine-readable code to calculate a charge for the services.

- In another aspect, the invention features method of billing for services, including determining a price plan that charges a client for services based on use, keeping a record of data of the client's use of the services, determining a charge for
25 the services based on the price plan and the record of the client's use of services and presenting a bill for the services.

- In an implementation, determining the price plan includes establishing the type of use for which the client desires to be billed, establishing a unit of cost for the type of use and generating a customized price plan template that reflects the type of
30 use and the unit of cost and is adapted to receive the record of data.

In another implementation, determining the charge for the services, comprises using the customized price plan template to calculate the service charge by processing the record of data based on the type of use and the unit of cost and presenting the bill for services includes generating a bill presentment template adapted to present the
5 record of data, the type of use, the unit of cost, and the service charge and presenting the bill presentment template to the client.

In still another aspect, the invention features a price plan network system, including at least one data collector adapted to collected client data from the network, a database coupled to the data collector, a price plan builder coupled to the database,
10 the price plan builder having a user interface a first software module having a price plan template adapted to direct a user through a series of steps to develop a price plan, a second software module coupled to the first software module, the second module including a plurality of components adapted to be interconnected to build a price plan source code, an object builder adapted to compile the source code into machine
15 readable code and a bill presentation module coupled to the database.

In an implementation, the system further includes a rater coupled to the database, adapted to be controlled by the machine-readable code to process the client data into billing data.

In another implementation, the bill presentation template includes a user
20 interface adapted to display the client data and the billing data

In yet another aspect, the invention features a price plan development and presentation system including means for building a price plan based on a client's use of services, means for collecting raw use-based client data, means for processing the use-based client data using the price plan and means for presenting the raw and
25 processed use-based client data to the client.

In another aspect, the invention features a network service provider method of charging a client for connecting the client to the network including developing a price plan agreement between the service provider and client, the agreement stating that the service provider provides a connection to the network in exchange for compensation
30 from the client, the compensation being calculated based on the client's use of the network, providing to the client a connection to the network, collecting use-based data

from the network, processing the use-based data based on the price plan and presenting the charges to the client.

In another aspect, the invention features a method of creating a price plan in a graphical program on a computer having a memory, a display, a user input and a processor, including storing in memory a plurality of executable functions
5 representing aspects of the price plan, assembling a data flow diagram on the display in response to user input to specify the price plan, the data flow diagram including function icons corresponding to the respective ones of the executable functions and generating an executable program from the data flow diagram.

10 In an implementation, the method further includes receiving price plan data in the executable program, processing the price plan data to calculate charge output data, creating a bill presentation template on a web page and providing the charge output data to the bill presentation template.

The details of one or more embodiments of the invention are set forth in the
15 accompanying drawings and the description below. Other features, objects, and advantages of the invention will be apparent from the description and drawings, and from the claims.

DESCRIPTION OF DRAWINGS

20 Fig. 1 illustrates a network access system

Fig. 2 illustrates an implementation of a billing system.

Fig. 3 illustrates a process flow chart of an implementation of a biller/rater.

Fig. 4 illustrates a process flow chart of an implementation of a graphical price plan.

25 Figs 5A-5C illustrate screenshots of an implementation of a price plan wizard.

Fig. 5D illustrates a flow chart of an implementation of price plan development using a price plan wizard.

Fig. 6A illustrates a flow chart for an embodiment of price plan development.

Fig. 6B illustrates a process flow chart of primitives used in a price plan.

Fig. 6C illustrates a screen shot of an embodiment of a price plan builder.

Fig. 7A illustrates a flow chart for an implementation of a diagnostic run time process.

Fig. 7B illustrates a flow chart for an implementation of a process to build
5 object code.

Fig. 8 illustrates a flow chart for an implementation of a rating engine process.

Fig. 9 illustrates a flow chart for an implementation of a billing presentation process.

Fig. 10A illustrates a screen shot of an implementation of a blank bill
10 presentation template.

Fig. 10B illustrates a screen shot of an implementation of a bill presentation template after it is provided with billing data.

Like reference symbols in the various drawings indicate like elements.

15 DETAILED DESCRIPTION

Fig. 1 illustrates a network access system 100. Several devices 110, such as personal computer stations, may desire access to a network 120, such as the Internet. A service provider 130, such as an NSP or ISP can provide network access through a series of connections 150, such as telephone modems or cable modems, linking the
20 devices 110 to the network 120. The devices 110 may desire to search the network 120 for information or gain access to any one of a variety of servers 140. The service provider 130 typically imposes a charge to link the devices 110 to the network 120. Therefore, the service provider 130 also collects data related to billing the devices 110 for connection to the network 120.

25 Fig. 2 illustrates an implementation of a billing system 200. A client 210 may want to access a network 220 such as the Internet. The client 210 can obtain this connection through a service provider 230. If necessary, a virtual private network (VPN) module 245 is used to provide a private connection between the client 210 and the network 230. The client 210 typically signs up with the service provider 230

through a registration/provisioning engine 240. Using the registration engine, the service provider 230 can query the client 210 for pertinent information concerning the client such as method of payment for services as well as a price plan that the client 210 desires to use. The client 210 can have a predetermined price plan that

5 determines how the client 210 is billed for the connection to the network 220. In a typical embodiment, the price plan is use-based. Use-based price plans are discussed in detail below. A collector 235 is used to collect the use-based data. The collector 235 is adapted to collect any of the use-based data, such as time-stamps for time

10 based-usage, bandwidth used for bandwidth-based usage, and number of emails and size of the emails for email-based usage. Typically, time stamps are collected for all types of data so that price per unit can be determined by time when use is incurred. For example, different price units can be applied on bandwidth use depending on the time of day. In other embodiments, the collector is adapted to collect any data

15 depending on the usage-based plan that the client 210 has chosen. Also, the collector 235 is used to manage the addresses of the numerous clients attached to the service provider 230, as well as providing application switches to the individual clients. In one embodiment, the collector utilizes Radius software to manage real time switching between clients to ensure that the proper use-based data is charged to the proper client. In another embodiment RMON2 software is utilized. The principal function

20 of the collector 235 is to collect and resolve usage data into accounts from various usage information sources such as RADIUS and RMON2 devices. The usage information sources are not part of the collector 235. The collector 235 does not typically have other functions such as managing client addresses, application switches and the like.

25 Referring still to Fig. 2, a biller/rater 250 is used to calculate the charges that the client 210 incurs while connected to the network 220. The biller/rater 250 is also used to present the bill to the client 210 after a billing cycle. Billing and rating functions of the biller/rater 250 are discussed in detail below. The biller/rater 250 is used in close conjunction with the collector 235. Once data is collected, the

30 management software is used to direct the appropriate billing charges the biller/rater 250 in order to calculate the correct charges which are billed to the client 210. The charges billed to the client 210 are dependent on the price plan for which the client

signed. An account management module 255 is used to manage use-based data collected from the client 210 through the collector 235. The rater 250 ensures that the client 210 is charged only for the network usage incurred by the client 210 and not a different client. The account management module 255 works in close conjunction
5 with the collector 235 and the biller/rater to store and manage the incurred charges. A market intelligence module 260 monitors and processes client 210 usage as well as price plans and client usage from other service providers on the network 220 to determine new marketing strategies. A payment/financial module 265 is used to track payments made by the client 210 and any other related financial matters. A database
10 270 is used to store client 210 information as well as price plans that the service provider 230 offers and a variety of other service provider-related information.

The client's 210 usage may be based only on time such as an hourly or monthly charge or may be based on some other usage-related basis such as bandwidth, content, email, storage web space, active internet protocol (IP) addresses,
15 downloaded movies, or downloaded music. For example, the client may be a server that receives a large volume of traffic on a daily business. A service provider 230 may want to charge a server-client with a large traffic volume by bandwidth in this situation. In another example, the client 210 may be billed only on the number of emails that are sent between the client 210 and the network 220. The client can have
20 a predetermined plan that outlines a how the client is charged for the network usage. Typically, charges are incurred by uses including but not limited to time, bandwidth, content, email, electronic storage, web-space used, multimedia usage such as video and audio downloads, and IP addresses in use. For example, a client may choose a plan that bills the client for time spent on the network. Under a time-based plan, the
25 client can be charged either by the hour or minute or can be charged once per week or month for unlimited minutes or hours. If the client is running a server on the network, the client may choose a plan that charges the client for the bandwidth used regardless of the time spent on the network. Under another plan, the client may be charged by the number of emails generated, whether sent or received. The email-based plan may
30 also include a further stipulation that the emails sent and received have to be of, at most, a certain size, or a further charge is incurred. For example, each email may be limited to 3 Megabytes, and for every Megabyte above 3 Mbytes, a further charge is

incurred to take into account additional bandwidth used for the email. Still another plan can allow a client to download movies or sound files from web sites at a certain price per movie or sound file. In one embodiment, the client can choose an omni-inclusive plan that encompasses several of the above-described plans. One goal of the omni-inclusive plan is to track charges for each of the plans included in the omni-inclusive plan. At the end of the billing cycle, the client can then be presented with a template including each charge of each plan. The client can then choose to pay for the plan that is the least expensive. In this way the client can use several plans on a trial basis and determine which plan suits the client the best. In an implementation of the omni-inclusive plan, the client is charged an additional cost each billing cycle for being able to decide among several plans. After several billing cycles, the client can opt out of the omni-inclusive plan and decide to remain with only one plan. For example, if an NSP offers several use-based plans such as time, bandwidth, email and web space, a client can choose to track charges for each of the plans under an omni-inclusive plan. At the end of a billing cycle, the client is presented with a bill that shows a charge for each of the time, bandwidth, email and web spaced used. The client can determine, for example, that the time-based charge was the least expensive and chooses to pay only the time-based charge. The client is also charged an additional cost to have the ability to choose among the plans. Thereafter, the client can then choose to be charged only under the time-based plan.

The billing and rating functions mentioned above are interrelated with one another and encompass both service provider 230 and client 210 interaction. Fig. 3 is a process flow chart illustrating an implementation of the operation of the biller/rater 250. The service provider 230 has the ability to graphically define a price plan with a series of software "primitives" which are connected together in a graphical programming environment, such as Visio®. Primitives and their functionality are discussed in detail below. A graphical price plan 310 is created by the service provider 230. The client 210 uses a client browser 330 in order to sign on to the network 220 and perform network functions such as web surfing through the service provider 230. The client browser 330 also is used to initially choose a price plan and to have the billing charges presented to the client 210 at the end of a billing cycle. Around the same time that the price plan is defined using the graphical price plan 310,

a bill presentation template 320 is also defined. The bill presentation template 320 is typically determined by the price plan chosen by the client 210. However, the bill presentation template can be modified to meet the needs of the client 210 or the service provider 230. As charges accrue, the collector 235 gathers the charge data and transfers it to a data repository 360. The charge data is transferred to a rating engine 340 and to the web bill presentment. The rating engine 340 is coupled to the graphical price plan 310. The rating engine 340 uses the price plan primitives from the graphical representation to process the charge data. Depending on the price plan chosen by the client 210, the data gathered by the collector may be different for different clients. For example, if the client 210 chose a time-based price plan, then the rating engine would typically acquire time stamp information from the gathered charge data. If the client's price plan were based on bandwidth used, then the rating engine 340 would acquire data indicating the number of bytes used by the client 210 during the network session. If the client's price plan were based on the number and size of the emails sent to and received from the network, the rating engine 340 processes only the email data.

The data repository 360 is also coupled to the web bill presentment 350 so that the client can also view the "raw usage" before they are processed by the rating engine 340. The web bill presentment 350 is the web interface with the client browser 330 for bill viewing. The rating engine 340 stores the processed use data in an account database 370, which is also coupled to the web bill presentment 350. The account database 370 stores all of the client's billing pertinent information such as a chosen price plan. The rating engine 340 is also coupled to a bill details database 380, which stores all of the processed data. Therefore, the client 210 can view and compare the raw use data with the processed use data. If the client 210 has chosen more than one price plan or all of the price plans as discussed above, the client is able to view all raw data and all processed data on the web bill presentment 350. The bill presentment 350 enables a client 210 to make a decision on which price plan works best for the client's needs.

30

Billing and Rating

The graphical price plan 310 is an aggregate of several components that the service provider 230 uses to build a price plan for the client 210. Fig. 4 illustrates a process flow chart of an implementation of the graphical price plan. In this

5 implementation, the aggregate of components that make up the graphical price plan are a rating plans database 410, a rating primitives database 420, and plan object builder 430 and a plan objects database 440. The rating primitives database 420 is a storage medium for the primitives that are the software components used to build a price plan. Primitives are programmed to serve a specific function. Primitives can be

10 programmed for virtually any function that is needed for the price plan. A description of primitives and their function is discussed below. A rating plans database 410 contains graphical programs made from interconnecting a series of primitives from the rating primitives database 420. The rating plans can be viewed as graphical source code. A plan object builder 430 is the compiler for the graphical source code.

15 The graphical source code is compiled and stored as an object file in a plan objects database 440. The rating engine 340 can access, from the plan objects database 440, a plan object that corresponds to a plan the client 210 has chosen. The rating engine 440 then executes the plan object code and processes the raw data from the collector 235.

20 While primitives are the fundamental building blocks of a price plan, building a price plan has several levels of complexity. For a non-technical person such as a marketing manager and salesperson, a high level editor is needed that allows a price plan to be developed quickly without requiring a high level of technical savvy. Figs. 5A-5C illustrate screenshots of an implementation of a price plan wizard. Fig. 5A is a

25 screenshot 500 of an implementation of an introductory screen for a price plan wizard. The screenshot displays textual instructions 505 to guide the developer through the price plan building process. Fig. 5B illustrates a screenshot 510 of an implementation of a screen in which a price plan developer can choose what kind of charges 515 incur in the price plan. As seen in the figure, a simple fixed charge 515a, such as one

30 monthly charge, can be implemented in the price plan. A recurring charge 515b such as a per minute charge can also be incurred. A use-based charge 515c can also be

incurred. The use-based charge 515c is the charge for such usages as bandwidth, email, and other usages as described above. Fig. 5C illustrates a screenshot of an implementation of a fixed charge screen 520. Using this screen, the developer of a price plan can impose a one-time charge on a client 210 for a set up charge or other
5 type of one-time charges. Using the price plan wizard, a developer can quickly customize a price plan for a customer without burdensome programming.

Fig. 5D illustrates a flow chart 530 for a typical embodiment of a price plan development using a price plan wizard. When the price plan wizard is initiated it is first determined 535 if there are any one-time charges. If there are, then the developer
10 provides 540 the details, that is, the cost and conditions of the one time charge. The next query is whether there are any recurring 545 charges. If there are recurring charges, then the developer provides 550, the amount and frequency of those recurring charges. Typically, the recurring charges will be a unit charge per minute, hour or month. The next query 555 is whether or not there are any use-based charges. However, in one implementation, there can be a recurring monthly charge as well as
15 an additional charge for bandwidth used, email generated and received or any other use-based charges as discussed above. If there are use-based charges, then the developer provides 560 the description and amount of the use-based charges. In a typical plan, there may be some free usage. Therefore it is queried 565 whether or not there is any free usage. If there is free usage, then the developer provides 570 the
20 details of the free usage. In one embodiment, the free usage may be a certain number of free connection hours. In another embodiment, the free usage may be use-based (i.e., bandwidth, email etc.) Another typical query 575 is whether there are any peak-related charges. If there are peak-related charges then the developer provides 580 the
25 details of the peak-related charges.

The price plan wizard then ceases the queries and a series of primitives are connected and compiled without interaction from the developer. Price plan object code is then available in the price plan database 410 for execution. The price plan graphical source code is typically stored as XML code.

While the high level price plan wizard is a vital tool for non-technical agents of the service provider 230, it lacks further the flexibility of directly connecting primitives which can provide further extensibility for price plan development.

As mentioned above, in one embodiment, a graphical programming program
 5 such as Visio® is used to build a price plan by interconnecting numerous primitives as a flow. In such an embodiment, a service provider agent can build a price plan quickly and efficiently using this price plan editor. For example, a marketing manager in an ISP company such as can quickly respond to changes that competitors have made in a price plan by opening the price plan wizard or having one of the
 10 programmers use a price plan editor to connect a series of primitives to match the price plan offered by a competitor. As another example, if a salesperson for an ISP is discussing a deal with a large corporation, the salesperson can quickly and efficiently make changes to a price plan using the price plan wizard. If the large corporation does not like any of the price plans that the salesperson can create using the price plan
 15 wizard, the salesperson can contact an engineer or programmer to use the primitive editor to create a more suitable price plan. The programmer can attempt to use the existing primitive in the rating primitives database 420 existing primitives from the rating primitive database 420. If the existing primitives are not sufficient, the salesperson can communicate the desires and needs of the corporation and the
 20 programmer can write a new primitive that meets the needs of the corporation. The flexibility and extensibility of primitive programming allows sales people, marketing managers and programmers to work independently or in conjunction to respond to marketing and sales needs in a competitive environment. In an embodiment, the primitive programming language can be any open platform language such as Java®,
 25 C, C++ and Visual Basic®.

Many references have been made to primitives. The discussion now addresses primitives and their functionality. The following table represents a subset of categories of a number of primitives and a short description of the function of each primitive as it can be used in any of several implementations:

30

Category	Primitive	Description
Accumulators	MonthlyAccumulator	Accumulate usage on a Monthly basis
	BillingCycleAccumulator	Accumulate usage on a billing cycle basis
	WeeklyAccumulator	Accumulate usage on a weekly basis
	DailyAccumulator	Accumulate usage on a daily basis
Filter	ThresholdFilter	Filter usage based on a threshold value
	DailyTimeFilter	Filter usage based on time of day
	WeeklyTimeFilter	Filter usage based on time of day and day of week
	UsageTypeSelector	Select usage type
	SpecialDaysTimeFilter	Filter based on designated days
	UsageTypeClassifier	Derive new usage type
Non-Usage Based	OneTimeCharge	Set up charge
	BillingCycleRecurringCharge	Recurring charges based on billing cycles
	MonthlyRecurringCharge	Monthly charge
Rater	StepRater	Rate based on steps having a multiplier and constant
	Adder	Sum all charges
Specialized Rater	CostResponsibilityMatrix	NetCountant Enterprise Cost responsibility matrix
	E164DialPlanRater	Calculate E.164 based phone charges
	TaxCalculator	Calculate applicable county/state/local taxes
	CurrencyConverter	Convert from one currency to another
	IptoLocationMapper	Map any given IP address to geographical location
Account Interface	AccountFinalizer	Post charges/credits to primary account
	SecondAccountFinalizer	Post charges/credits to a second account

Table 1.

Primitives can be programmed directly in a primitive programming environment, which typically involves adding primitives and interconnecting primitives using typical data flow programming. If there are no primitives that meet
5 the needs of a particular price plan, a programmer can create a new primitive by traditional programming methods such as Java®, C, C++ and Visual Basic®.

Primitives are also connected while using the price plan wizard. As discussed above, using the price plan wizard, the developer chooses the price plan options in the price plan wizard screens. During this development, a price plan generator, which is
10 part of the price plan wizard, is connecting a number of primitives together according to the choices made in the price plan wizard. Either way, primitives are fundamental building blocks of the price plan.

Fig. 6A illustrates a flow chart of an implementation of graphical programming using primitives to develop a price plan. Similar to the price plan
15 wizard described with respect to Figs. 5A – 5C, it is first determined 605 whether there are any one time charges. If there are, the necessary primitives are added 610 in the programming editor to reflect the one time charge. It is next determined 615 if there are any recurring charges. If there are recurring charges, primitives are added 620 to reflect these charges. It is next determined 625 if there are any use-based charges (as discussed above). Once again, if there are any use-based charges,
20 primitives are added 630. If there is any free usage 635, the appropriate primitives are added 640. There may be a charge difference for on-peak and off-peak usage 645, and primitives are appropriately added 646 to reflect those differences in charges. Simple adding primitives and account management primitives are also added 647.

25 Fig 6B illustrates an implementation of a price plan developed by interconnecting primitives using a price plan editor. As discussed above with respect to Fig. 4, the primitive programming environment is the graphical source code that the developer uses to ultimately create a plan object for execution by the rating engine 340. The example used in Fig. 6 is a simple price plan that consists of a \$10 one-time set-up charge and a single monthly charge of \$20 with monthly billing at \$20/month.
30 The plan includes 100 free connection hours. After 100 hours peak time (7PM-

midnight), the plan bills at \$1/hr. In addition, off-peak time (midnight – 7PM) is billed at \$0.50/hr for the first additional 100 hours, then \$0.25 thereafter.

Each primitive in the price plan is programmed with the necessary data as described. A OneTimeCharge primitive 655 is programmed to incur a set-up charge of \$10 only the first time that the corresponding plan object code is executed, and includes software logic to prevent multiple recurrences of the one time charge. Thereafter, the OneTimeCharge primitive 655 does not execute. Running in parallel to the OneTimeCharge primitive 655 is a RecurringCharge primitive 660 that incurs the charge of \$20 for every passing month. The RecurringCharge primitive 660 uses the data gathered by the collector 235 in order to obtain time stamp data 651 to determine when a time period of one month has occurred. A FreeStepRater 665 is a primitive programmed to “step up” the numbers of hours up to a specified number of hours. It is programmed with data 667 in order to cease stepping, in this example, at 100 hours. Two DailyTimeFilter primitives 670, 679 are used to allow the program data to flow one way or another. If the data is gathered during peak hours of 19:00 to 24:00, then the data is permitted to flow through the DailyTimeFilter 670. Data 671 is programmed into the DailyTimeFilter 670 to specify the time range that data is able to flow. During these peak hours data does not flow through the DailyTimeFilter 675 which is programmed with data 676 that only allows data flow during the hours of 00:00 and 18:59. A corresponding PeakStepRater primitive 680 is programmed to determine the cost of usage during the peak hours. As discussed above, the price plan in this example charges \$1/ hr of peak time usage. Multiplier data 681 is programmed into the PeakStepRater primitive 680. These charges only apply after the 100 free hours have passed as set by FreeStepRater 665. Similarly, if the client 210 is accessing the network during off peak hours, a charge of \$0.50/hr applies if 100 hours of off-peak time has not yet been attained, and a charge of \$0.25/hr applies if the number of off-peak hours exceeding 100 has been attained. However, if at the FreeStepRater 665, it is determined that the number of hours has not exceed the initial 100 hours (whether peak or off peak), the usage data is caused to be blocked from the DailyTimeFilters 670, 675 and the StepRaters 680, 685 and no cost data flows directly to an adder 690 through these paths because the FreeStepRater 665 will have no output to trigger the succeeding primitives. The adder 690 is programmed to add all

of the charge data that is available to it. After the Adder 690 executes, the data flows to an Account Finalizer primitive 695 that transfers the data to the account management module 255 or the account database 370. The charge data at this time is \$0 unless it is the first time the client 210 is using the account, and the one time charge is incurred from the OneTimeCharge primitive 655.

In the above-described graphical programming implementations, several primitives are used. Some primitives are executed in parallel when data is available to them. However, for many programming sequences, primitives can be required to run in series. The systems and techniques described are fault tolerant so that one primitive does not execute until it has the data it needs either from the collector or from the output of another primitive. This fault-tolerance prevents errors from occurring in a situation where a primitive tries to execute before it has its proper data. The systems and techniques described imply a proper execution sequence of the primitives to make sure that input data is available from the preceding primitive before the current primitive is executed. This can be accomplished by performing a topological sort on the price plan graph treating the primitives as nodes and interconnections as directed edges. For example, in the price plan example described above with respect to Fig. 6B, the Adder primitive 690 will not execute until it has all of the data from the OneTimeCharge 655, RecurringCharge 660, PeakStepRater 680 and OffPeakStepRater 685 primitives. If there is no data available from those primitives, such as in the case where there is no longer a one time charge, an appropriate signal such as a "0" null data is sent from the primitive. In one embodiment, primitives are graphical icons used in data flow programming. In this embodiment, building a price plan creates source code for a graphical programming environment such as Visio®.

Fig. 6C is a screen shot of an implementation of a price plan builder using a Visio® environment. In this implementation, the Visio® environment is used to develop the graphical programming necessary to build the price plan. A series of primitives 697 are placed in an editing field 699 and connected with a series of data flow connectors 697a. Palette tools 696 are used to build the graphical program.

Also shown in the screen shot is an informational field 698 that is used to provide information about a highlighted primitive.

Primitives can also be created for a variety of purposes such as monitoring the incoming data from the collector 235 to make sure that the client 210 is not using the network connection for any unauthorized purposes. For example, if the client 210 is paying only for time-based usage, then it is inappropriate to connect a server and use a disproportionate amount of bandwidth. An implementation of a primitive can check the amount of bandwidth a client 210 is using and compare it with the price plan.

In order to assure accuracy of the price plan and ultimate billing, the price plan wizard or a similar template can be used to test the price plan with known data. Price plan object code is then available in the price plan database 410 for execution. When retrieved for execution the developer can run the code in a run time diagnostic environment with known data with knowledge of charges the result. In this way, the developer can run diagnostic data to check the accuracy of the code before it is compiled to object code and used in an actual billing environment.

Fig. 7A illustrates a flow chart of an implementation of diagnostic data process. The code stored in the price plan database 410 is retrieved 705. The developer then determines 710 if there are any modifications necessary in the code. In the first instance of retrieving the code, there are typically no modifications that are necessary and the developer has made the determination that the test code is to be tested 715. The developer can then use the testing run time environment to test the source code with known data by stepping 720 through the code. Thereafter, the developer can determine 710 once again if there are any modifications to be made. If there are modifications, then the developer makes those modifications. Typical modifications can be but are not limited to adding 730 a new processes (or primitives), modifying 735 existing processes (existing primitives can be modified but involve more complicated software programming) and modifying 740 the connections between processes (primitives). Typical modifications can be but are not limited to adding 730 a new process (or primitive), modifying 735 the property (or data) with existing processes and modifying 740 the connections between primitives.

Once a price plan has been built and tested, it is stored in the rating database 410. In a typical embodiment, the price plan is stored as extensible markup language (XML). When a price plan is programmed and stored, it is subsequently retrieved and compiled into object code for execution by the rating engine 340. Fig. 7B illustrates a flow chart of an implementation of a price plan object builder. To compile the graphical source code, the XML price plan code is obtained 755 from the plan database 410. The XML code is parsed 760 to create an object representation of the code. At this time it is determined 765 if there are anymore processes that are to be in the plan. If more processes are desired in the plan, the developer can find and load 770 the primitives from the primitive database 420 and object representations of those primitives are created 775. Properties of the object primitives are established 780 such as by furnishing specific data (i.e., constants or multipliers are indicated in Table 1.) If there are no more rating processes to be added to the plan, then the compilation process establishes 790 the interconnections between the primitives. At this point, the fault tolerance discussed above is built into the code. The execution sequence of the rating primitives as discussed above is determined. Each primitive is programmed to "know" the data for which it has to wait before executing. The plan semantics are then validated 795, for example, any specific provisions in the plan. The object code is then delivered 796 to the rating engine 340 for execution.

Fig. 8 illustrates a flow chart for an implementation of a rating engine 340 object code execution process. When the code is executed, the rating engine 340 is told to retrieve 805 a usage record. This record is the usage data gathered by the collector 235. The rating engine 340 then retrieves 810 usage record account information from the account management module 255 or the account database 370 in order to determine what plan is to be used. The rating engine 340 then retrieves 815 the plan ID for a record and date. The plan logistics may differ for different usage records and dates. The rating engine 340 determines 820 if the object code identified by the plan ID is available in a code cache for execution. If the object code is not available, then the proper primitive connections are established 850 (same as establishing 790 primitive interconnections in Fig. 7B). If the object code is not available, then the object code is constructed and delivered (Fig. 7B). Once the code is available, then the code is initialized 825 with the pertinent variables from the

account information (i.e., rates, free-usage etc.) The usage record is then rated 830 appropriately based on the plan and the account variables are updated 835. The bill data is then output 840 to the bill details database 380. This process is repeated for all data records retrieved by the collector 235.

5

Bill Presentation

Referring again to Fig. 3, the bill presentation template 320 and the web bill presentment are typically embodied, respectively, in a web page delivered to the client's browser 330. In one embodiment, a bill presentation web page (not shown) is typically created and modified by using a HyperText Markup Language (HTML) editor. The HTML editor allows the service provider 230 and/or the client 210 to create a web page that best meets the needs in bill presentation. The service provider 230 provides elements that are important to the bill presentation process such as the separate charges incurred during the billing cycle. The client 210 can optionally add billing data and the order in which it is presented. For example, the client can present the incurred charges and the raw usage-data side by side to check on the accuracy of the charges. The client can also add the terms of the billing plan as a reminder of how charges are incurred. If the client 210 is signed up to use multiple billing plans, the charges for all the plans can be displayed side by side to provide a price comparison so that the client can ultimately choose the price plan of choice.

Fig. 9 is a flow chart of an implementation of a bill presentment process. When the client 210 makes a request to view billing information on the client browser 330, the service provider 230 sends 905 a web page, as an HTML file, to the client's browser 330. The client can, at anytime during the process, for example, make 935 modifications to the web page to customize the page to meet the client's 210 needs. The service provider 230 provides 910 the billing data from the billing details database 380. If there is a network connection, the service provider 230 can also provide 915 any real time data as the rating engine 340 produces it. The HTML file is then formatted 920 to display the data to the client 210. As the client 210 is viewing the web page, it is determined 925 whether or not there is more real data available. If there is more real time data, then it is provided 915 to the client's browser 330. The

process is repeated as necessary. The information is all output 930 as an HTML file for viewing by the client 210.

Fig. 10A is a screen shot of an implementation of a bill presentation template 1000. This template is used as a development tool for the actual bill presentation that the client 210 sees. The figure illustrates that the template 1000 is displayed on a typical web page 1005. The template 1000 contains various fields as described above. A field 1010 for a one time installation cost is first shown. This field will typically be present only the first time the template 1000 is displayed to the client 210. In this particular price plan, the client 210 is using a price plan that includes charges for web space used and email. Respective fields 1015, 1020 are displayed in the template 1000 to reflect these charges. Fields 1025, 1035 are displayed for on and off peak charges. Respective fields 1030, 1040 are displayed reflecting the number of hours connected for on and off peak. Further charges present in the billing plan are an FTP data transfer charge and a Voice over IP data charge. Respective fields 1045, 1055 reflect these charges, and fields 1050, 1060 reflect the amount of data transferred to incur those charges.

Fig. 10B illustrates a screen shot of an implementation of the bill presentation template 1000 after it has been filled with billing data. The template 1100, as seen by the client 210, contains a field 1110 for the billing period, as well as a field 1120 displaying the total costs. The fields 1130 as discussed with respect to Fig. 10A are filled with the actual billing data. Several additional fields can be added to reflect other charges that are incurred using other price plans. Figs. 10A and 10B are used only as examples of the types of templates that can be used in both bill presentation development and an actual bill presentation.

The techniques and apparatus discussed above have several business applications. In a typical application, as discussed above, price plans are developed by an ISP or NSP to meet the needs of different types of clients with different needs. In addition, the ISP or NSP will be able to achieve maximum charges from the clients commensurate with their use of the network. For example, if a client is attaching a server that uses a lot of bandwidth, the ISP or NSP is able to detect that used bandwidth and make an appropriate charge.

In another embodiment, the customer accounts can be partitioned according to account numbers. For example, if there are one million customer accounts, 10 partitions can be created by assigning account numbers 1 to 100,000 to the first partition, 100,001 to 200,000 to the second partition and so forth. A rating engine is then assigned to each partition. The rating engine selects and rates usage records with the assigned account number range. Scalability is achieved by creating the proper number of partitions with corresponding rating engines and distributing the rating engines to multiple computers. By assigning account number ranges to rating engines instead of having rating engines to process arbitrary accounts, caching of account information is possible (since no other modifications are made to the same account) and thereby increase the overall usage record processing rate.

In another embodiment, more than one engine can be assigned to work on the same account partition. In such a case, each engine working on the same partition is assigned a unique priority. The rating engine with the highest priority is active processing usage records. Other engines are monitoring the status of the active engine. In the event that the monitoring engines detect inactivity of the primary engine, the next highest priority engines takes over processing the usage records. The active engine updates a record in the database with a current time stamp. IF the time stamp fails to be updates, the active engine is assumed to be non-functional. A monitoring engine initiates a take over procedure. When the failed engine recovers, it regains control from a lower priority engine and resumes processing of the usage records. No central controlling entity is involved in the fail other than the database. Each engine operates autonomously.

Various aspects of the techniques and apparatus may be implemented in digital circuitry, or in computer hardware, firmware, software, or in combinations of them. Apparatus of the invention may be implemented in a computer products tangibly embodied in a machine-readable storage device for execution by a programmable processor. The foregoing techniques may be performed, for example, by a programmable processor executing a program of instructions to perform functions of the invention by operating on input data and generating output. The methods may advantageously be implemented in one or more computer programs that

are executable on a programmable system including at least one programmable processor coupled to receive data and instructions from, and to transmit data and instructions to, a data storage system, at least one in/out device, and at least one output device. Each computer program may be implemented in a high-level

5 procedural or object-oriented programming language, or in assembly or machine language if desired; and in any case, the language may be compiled or interpreted language. Suitable processors include, by way of example, both general and special purpose microprocessors. Generally, a processor will receive instructions and data from read-only memory and/or random access memory. Storage devices suitable for

10 tangibly embodying computer program instructions and data include all forms of non-volatile memory, including by way of example, semiconductor devices, such as EPROM, EEPROM, and flash memory devices; magnetic disks such as internal hard disks and removable disks; magneto-optical disks; and CD-ROM disks. Any of the foregoing may be supplemented by or incorporated in, specially designed application-

15 specific integrated circuits (ASICs).

A number of embodiments of the invention have been described. Nevertheless, it will be understood that various modifications may be made without departing from the spirit and scope of the invention. Accordingly, other embodiments are within the scope of the following claims.

20

What is claimed is:

1. A method of determining a billing-cycle price for network services,
comprising:
 - 5 using a plurality of graphical elements on a display to establish a price plan for each of a plurality of clients, wherein the price plan sets a price per use unit;
collecting use data from the plurality of clients;
sorting the use data into data groups corresponding to each of the plurality of clients; and
 - 10 using the graphical elements to process the data groups to determine the billing-cycle price for each of the plurality of clients, wherein the billing-cycle price is calculated from use data and the price per use unit.
2. The method of claim 1 wherein the price per use unit is determined by time
15 that each of the clients used the network services.
3. The method of claim 1 wherein the price per use unit is determined by network bandwidth that each of the clients used.
- 20 4. The method of claim 1 wherein the price per use unit is determined by network content that each of the clients used.
5. The method of claim 1 wherein the price per use unit is determined by email that each of the clients generated on the network.
25
6. The method of claim 1 wherein the price per use unit is determined by electronic storage space used by each of the clients.

7. The method of claim 1 wherein the price per use unit is determined by web space that each of the clients used.
8. The method of claim 1 wherein the price per use unit is determined by
5 multimedia usage that each of the clients used.
9. The method of claim 1 wherein the price per use unit is determined by Internet protocol addresses in use by each of the clients.
- 10 10. The method of claim 1 further comprising:
allowing each of the plurality of clients to choose a price plan that includes
more than one price per use unit;
calculating a billing-cycle price for each price per use unit;
presenting each billing-cycle price to each of the clients; and
15 allowing the client to choose the billing-cycle price that is the most cost
efficient.
11. The method of claim 1 further comprising:
comparing a cost associated with the plan to the billing cycle price;
20 determining a profit of each of a plurality of price plans; and
determining an action to increase the profit.
12. The method of claim 11 wherein the action is changing one or more of the
plurality of clients to a new price plan.
- 25 13. The method of claim 11 wherein the action is eliminating one or more of the
price plans.

14. The method of claim 1 further comprising:
developing a new price plan; and
using a set of old use-based data in the new price plan to determine an effect
5 on profits.
15. The method of claim 1 further comprising:
transmitting a bill presentation template to each of the clients.
16. The method of claim 15 wherein the bill presentation template is transmitted
10 to a client web browser connected to the network.
17. The method of claim 16 wherein the bill presentation template displays
charges incurred during the billing cycle.
- 15 18. A method of developing a price plan for services on a display, comprising:
providing an interface on the display;
providing on the interface images adapted to receive price plan criteria; and
receiving in the images price plan criteria.
- 20 19. The method of claim 18 wherein images on the interface are graphical user
interface (GUI) screens.
20. The method of claim 19 wherein providing the images comprises presenting
the GUI screens in an order to present price plan criteria input options for generating
25 the price plan to the user.
21. The method of claim 20 further comprising:

generating the price plan into source code;
compiling the computer code into machine-readable code.

22. The method of claim 21 further comprising:

5 supplying the price plan data to at least one of the source code and the
machine-readable code to calculate a charge for the services.

23. The method of claim 18 wherein the images on the interface are graphical
elements representing components of the price plan.

10

24. The method of claim 23 wherein providing the images comprises:

arranging the graphical elements on the display that graphically represents the
calculations for the price plan charge; and

15 providing interconnections among the graphical elements to provide price plan
data flow paths.

25. The method of claim 24 further comprising:

storing the arrangement of graphical elements as source code; and
compiling the source code into machine-readable code.

20

26. The method of claim 25 further comprising:

supplying the price plan data to at least one of the source code and the
machine-readable code to calculate a charge for the services.

25 27. A method of billing for services, comprising:

graphically determining a price plan that charges a client for services based on
use;

keeping a record of data of the client's use of the services;

determining a charge for the services based on the price plan and the record of the client's use of services; and

presenting a bill for the services.

5

28. The method of claim 27 wherein determining the price plan comprises:

establishing the type of use for which the client desires to be billed;

establishing a unit of cost for the type of use; and

generating a customized price plan template that reflects the type of use and

10 the unit of cost and is adapted to receive the record of data.

29. The method of claim 28 wherein determining the charge for the services, comprises using the customized price plan template to calculate the service charge by processing the record of data based on the type of use and the unit of cost.

15

30. The method of claim 28 wherein presenting the bill for services comprises:

generating a bill presentment template adapted to present the record of data, the type of use, the unit of cost, and the service charge; and

presenting the bill presentment template to the client.

20

31. A price plan network system, comprising:

at least one data collector adapted to collect client data from the network;

a database coupled to the data collector;

a price plan builder coupled to the database, the price plan builder having:

25

a user interface;

a first software module having a price plan template adapted to direct a user through a series of steps to develop a price plan;

a second software module coupled to the first software module, the second module including a plurality of components adapted to be interconnected to build a price plan source code;

an object builder adapted to compile the source code into machine readable code; and

a bill presentation module coupled to the database.

32. The system of claim 31 further comprising a rater coupled to the database, adapted to be controlled by the machine-readable code to process the client data into billing data.

33. The system of claim 32 further comprising at least one additional rater, wherein the rater and the at least one additional rater are adapted to operate autonomously of each other, and further adapted to monitor each other for failure, and to operate in the event that one of the rater and the at least one additional rater fail to operate.

34. The system of claim 32 wherein the bill presentation template comprises a user interface adapted to display the client data and the billing data.

20

35. A price plan development and presentation system comprising:
means for building a price plan based on a client's use of services;
means for collecting raw use-based client data;
means for processing the use-based client data using the price plan; and
means for presenting the raw and processed use-based client data to the client.

25

36. A network service provider method of charging a client for connecting the client to the network comprising:

developing a price plan agreement between the service provider and client, the agreement stating that the service provider provides a connection to the network in exchange for compensation from the client, the compensation being calculated based on the client's use of the network;

- 5 providing to the client a connection to the network;
collecting use-based data from the network;
processing the use-based data based on the price plan; and
presenting the charges to the client.

- 10 37. A method of creating a price plan in a graphical program on a computer having a memory, a display, a user input and a processor, comprising:
storing in memory a plurality of executable functions representing aspects of the price plan;
assembling a data flow diagram on the display in response to user input to
15 specify the price plan, the data flow diagram including function icons corresponding to the respective ones of the executable functions; and
generating an executable program from the data flow diagram.

38. The method of claim 37, further comprising:
20 receiving price plan data in the executable program;
processing the price plan data to calculate charge output data.

39. The method of claim 38 further comprising:
creating a bill presentation template on a web page; and
25 providing the charge output data to the bill presentation template.

1/17

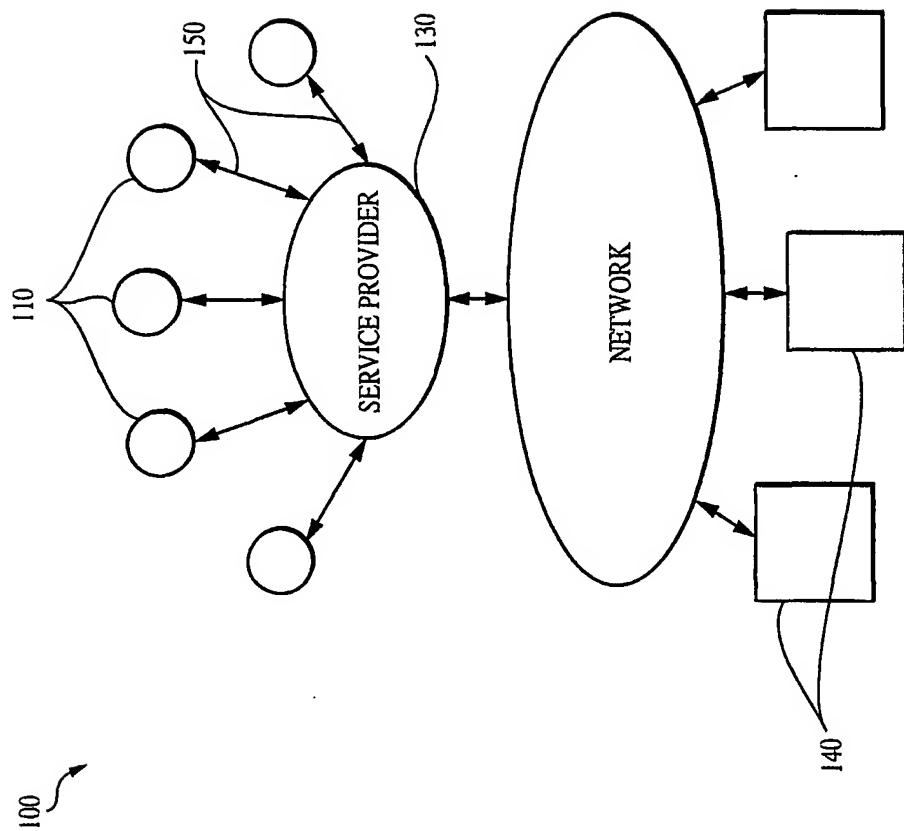


FIG. 1

2/17

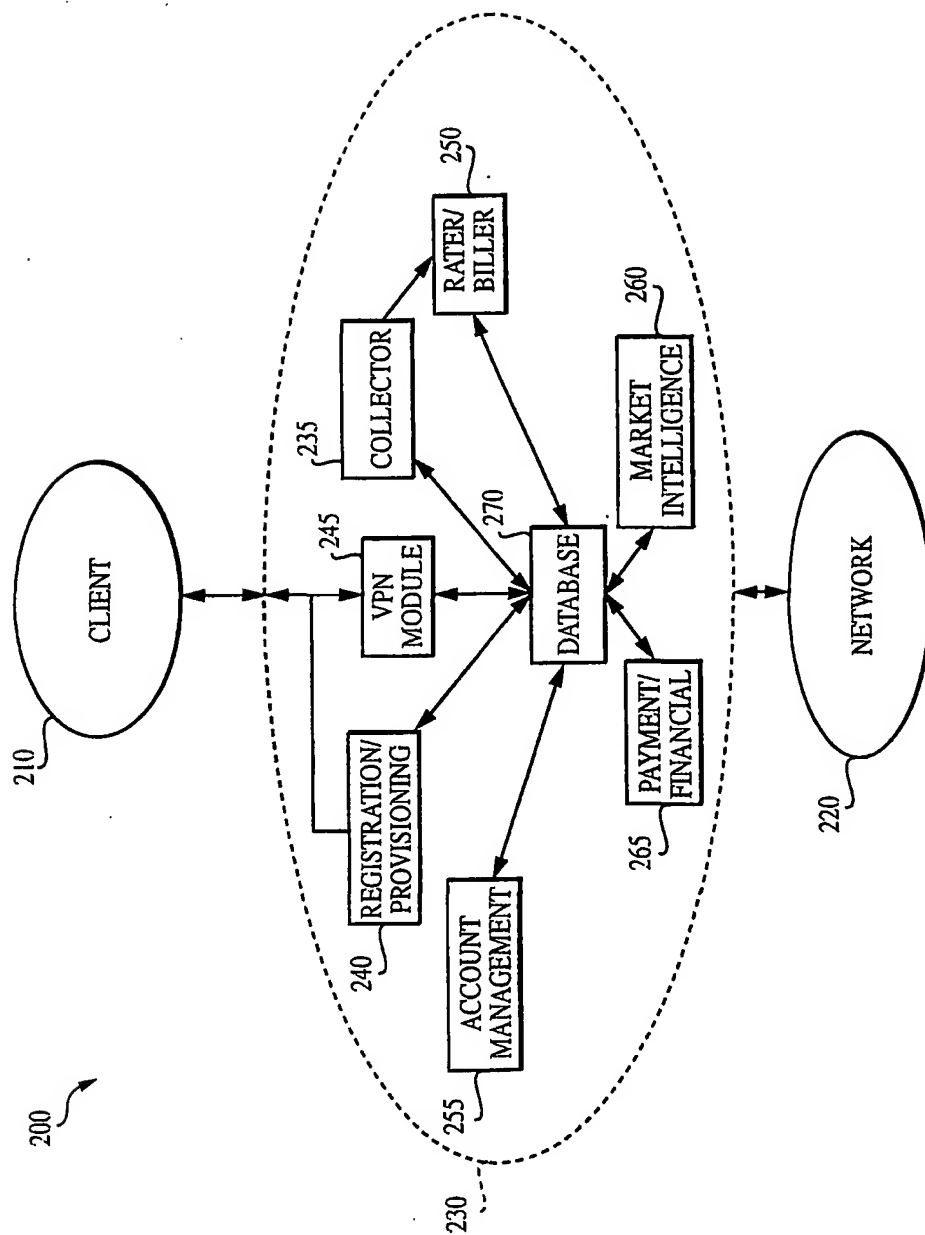


FIG. 2

3/17

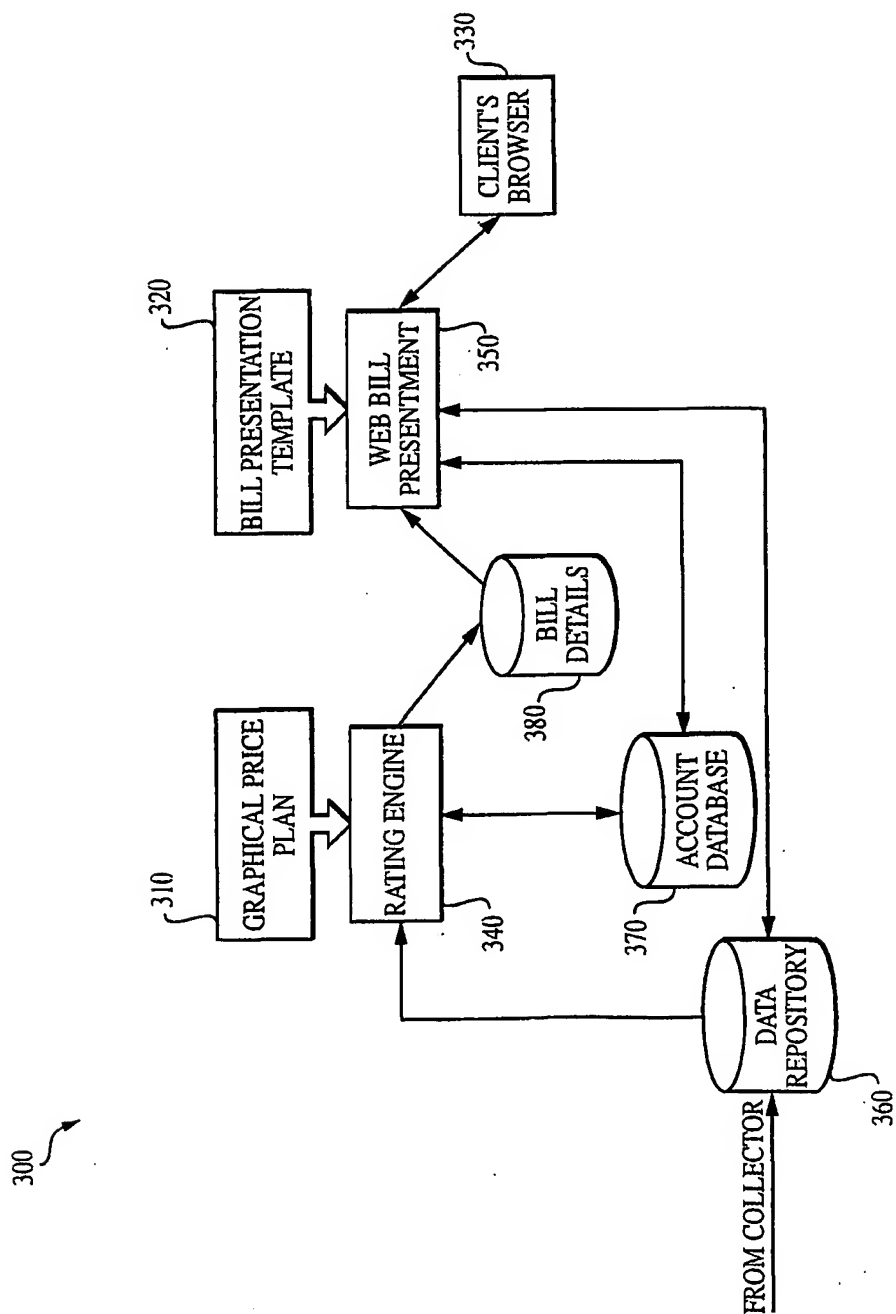


FIG. 3

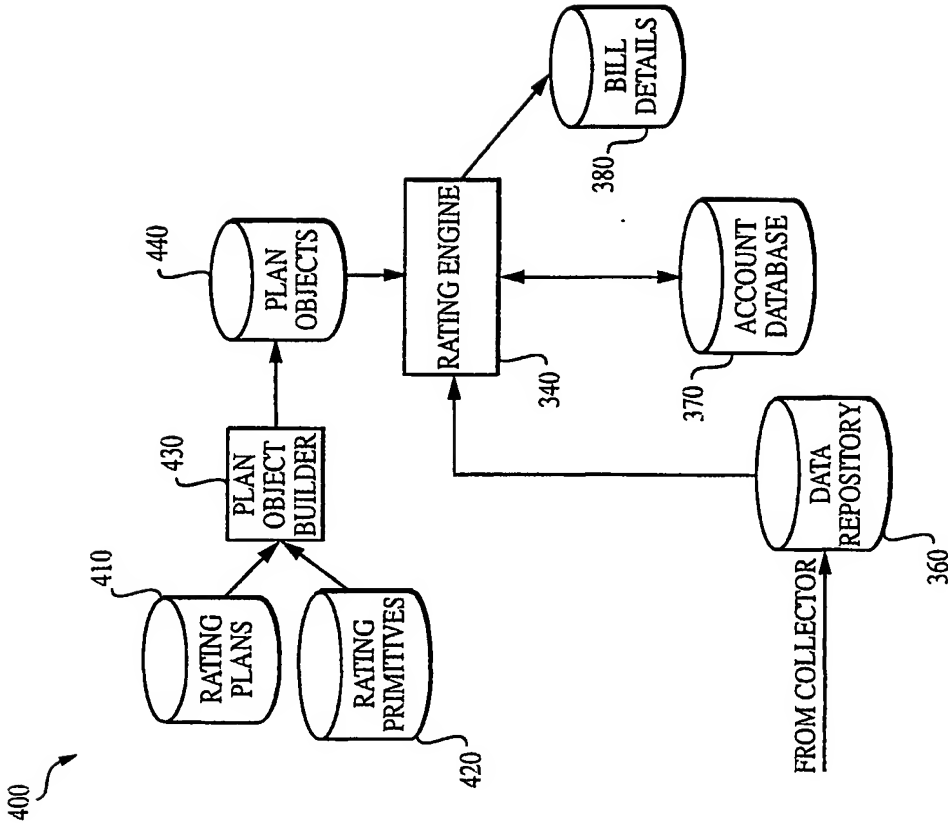


FIG. 4

500

5/17

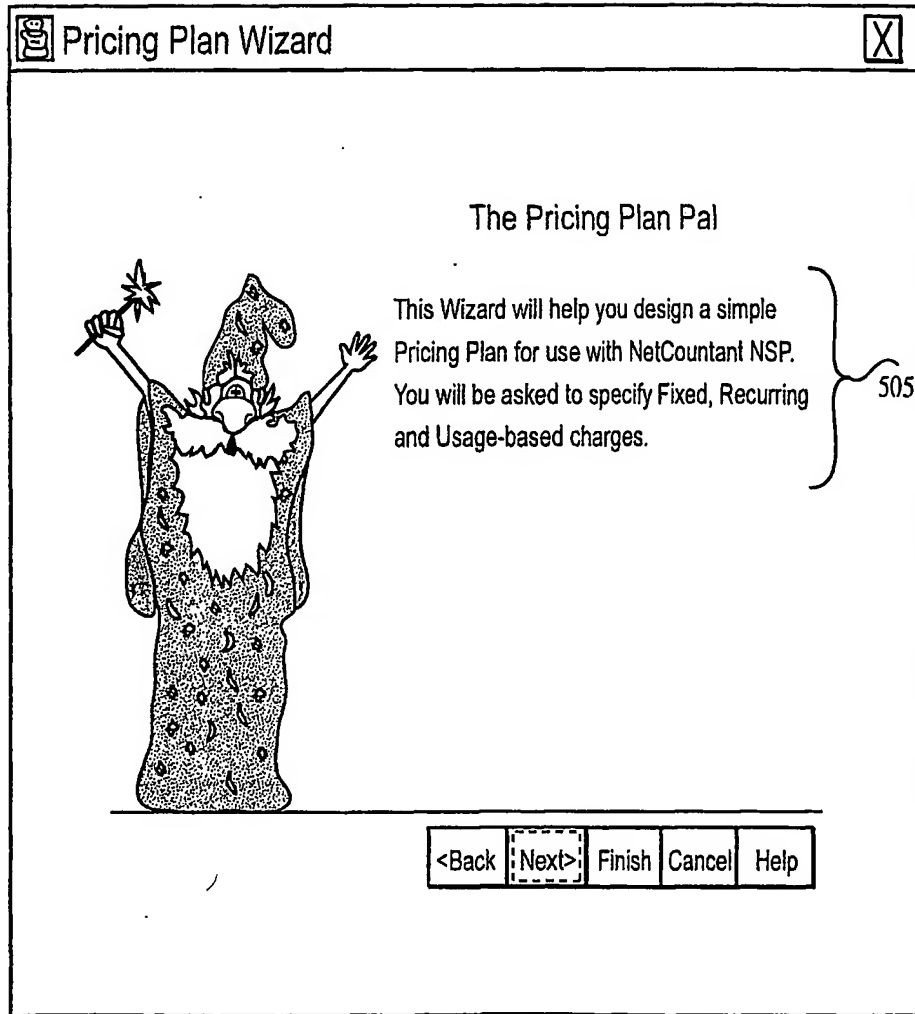
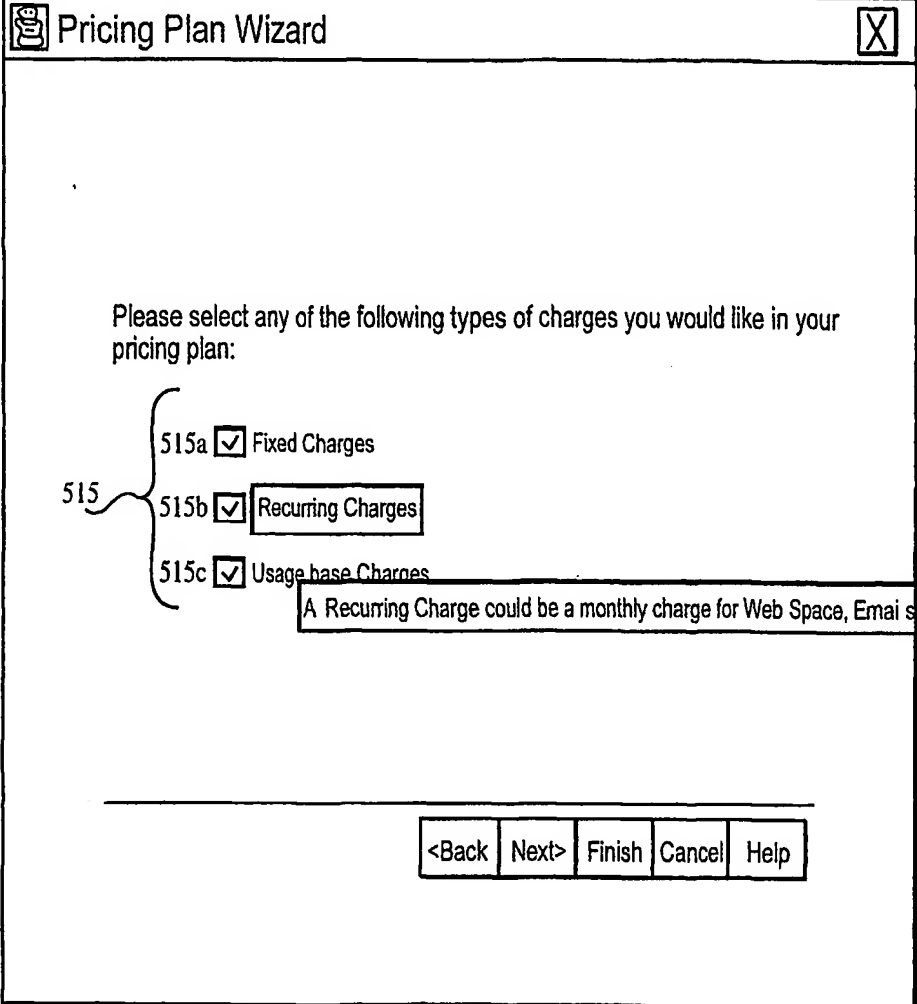




FIG. 5A

510


6/17



 Pricing Plan Wizard 

Please select any of the following types of charges you would like in your pricing plan:

515 { 515a ☒ Fixed Charges
515b ☒ Recurring Charges
515c ☒ Usage base Charges

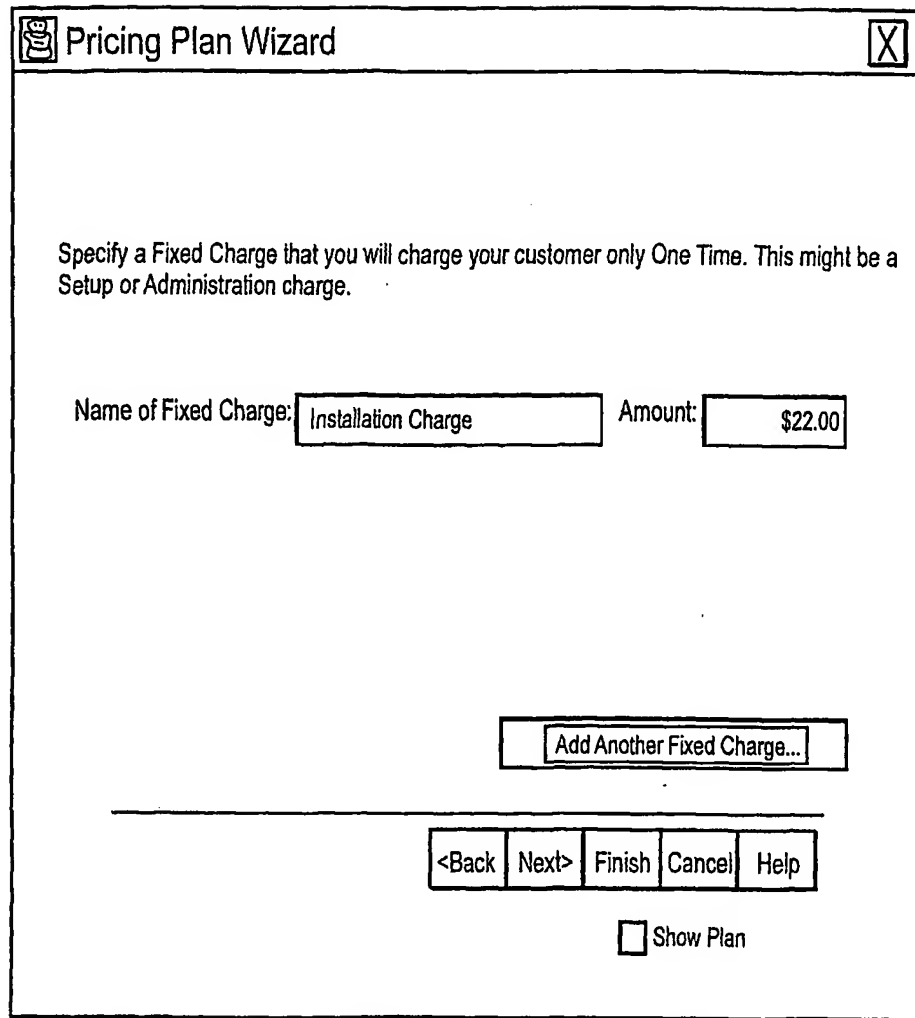
A Recurring Charge could be a monthly charge for Web Space, Email s

<Back Next> Finish Cancel Help

FIG. 5B

520

7/17



The image shows a software dialog box titled "Pricing Plan Wizard". It contains a text instruction, two input fields for a fixed charge, a button to add another charge, a set of navigation buttons, and a checkbox.

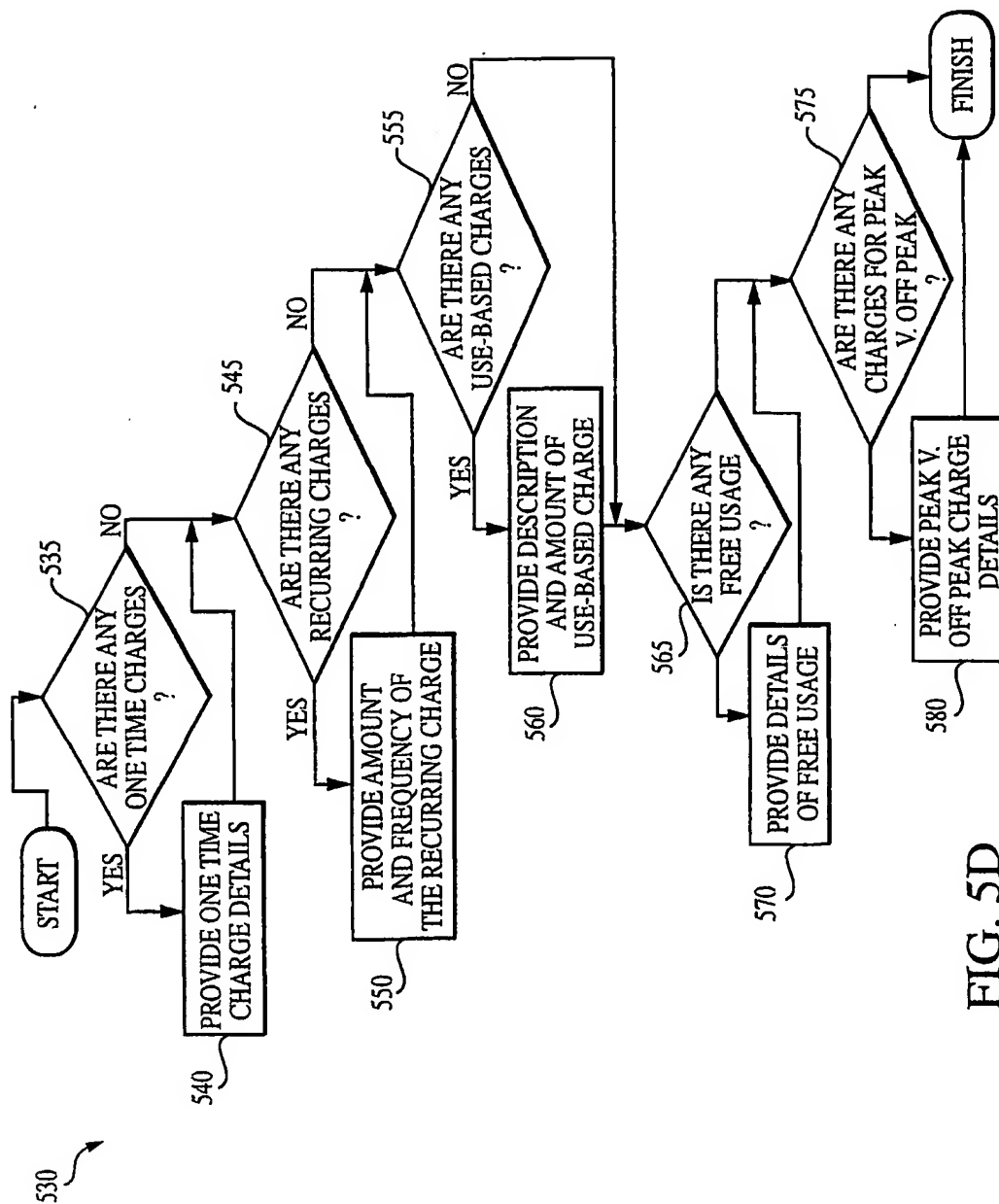
Specify a Fixed Charge that you will charge your customer only One Time. This might be a Setup or Administration charge.

Name of Fixed Charge: Amount:

☐ Show Plan

FIG. 5C

8/17



9/17

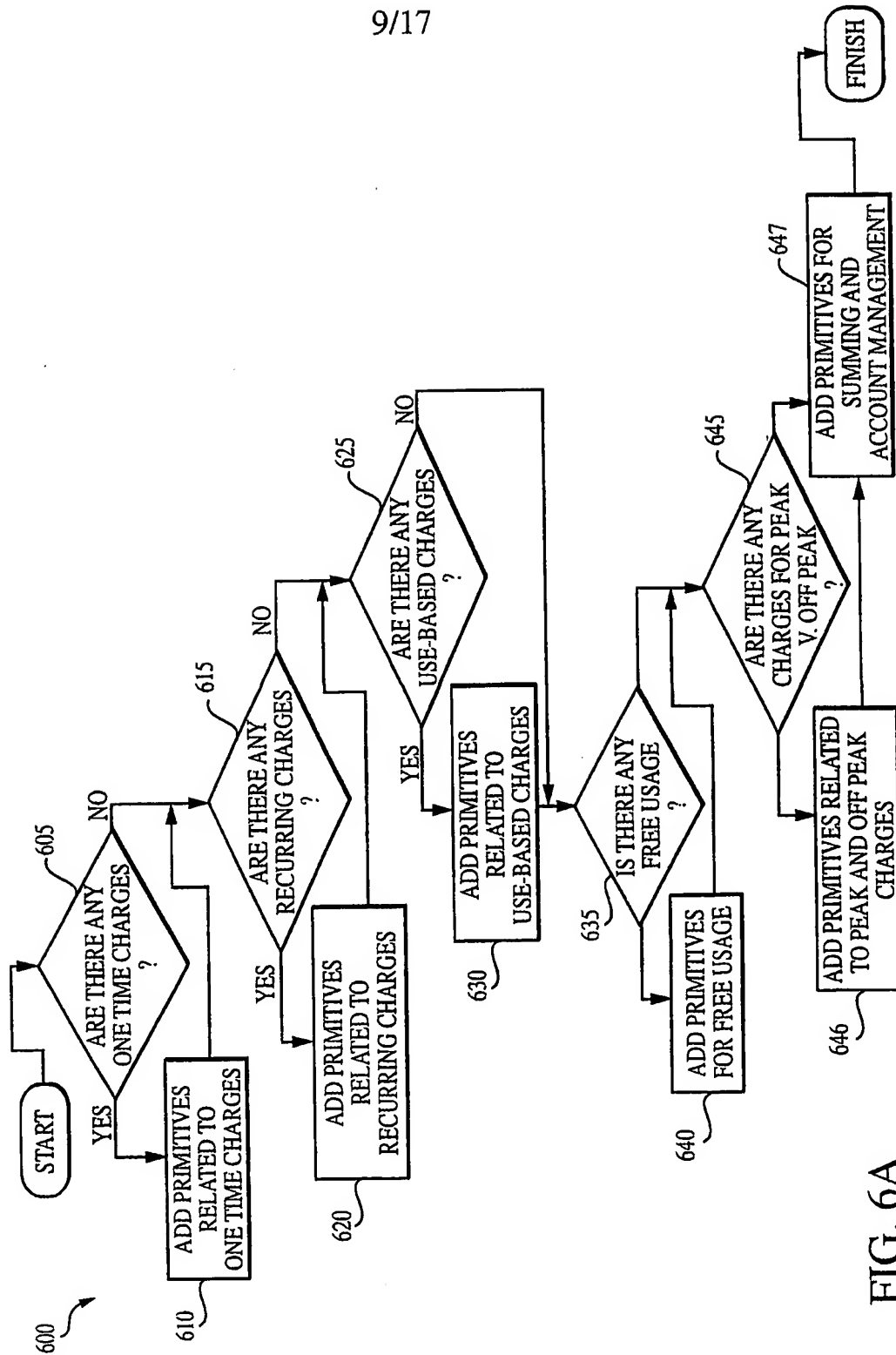


FIG. 6A

10/17

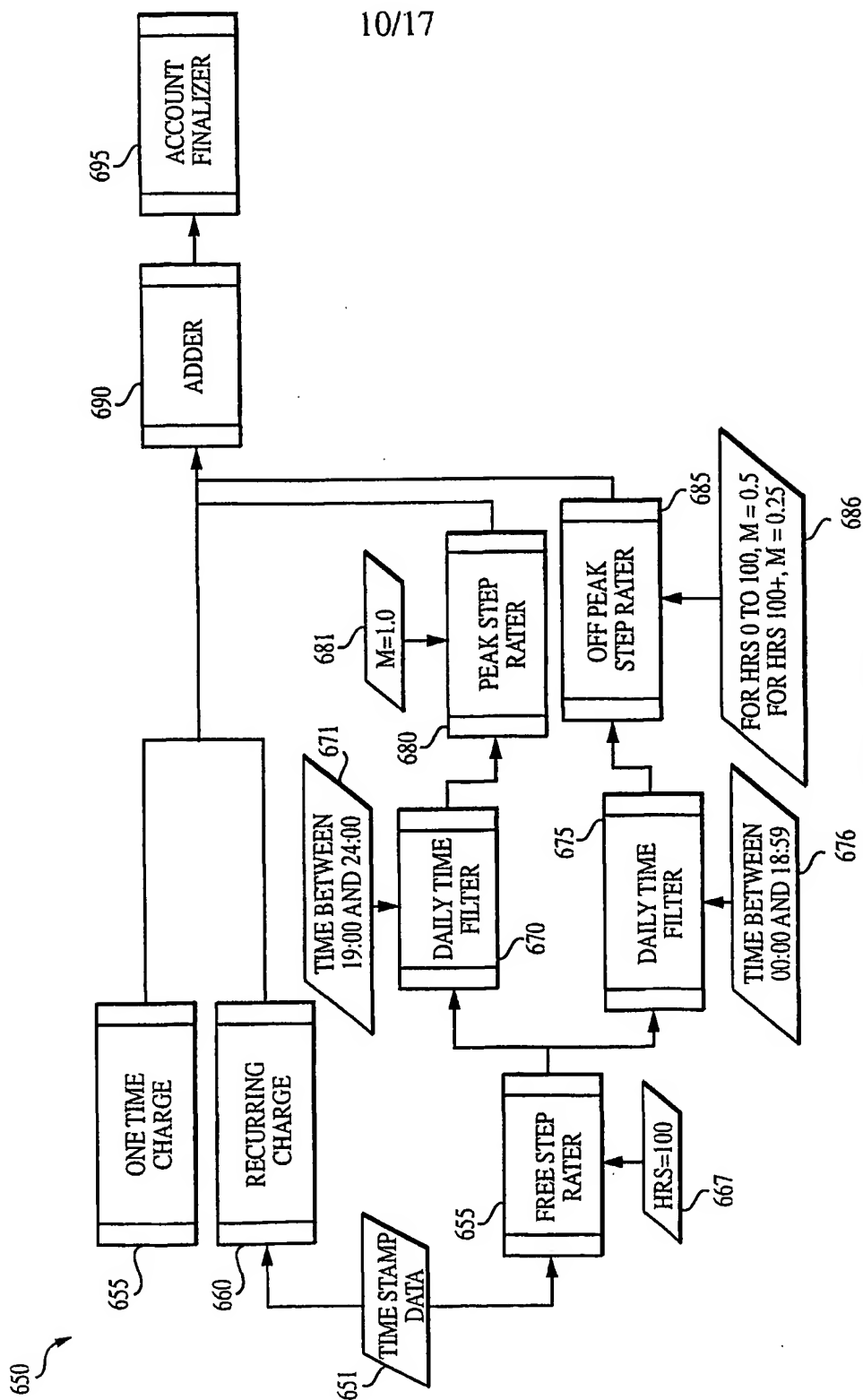


FIG. 6B

11/17

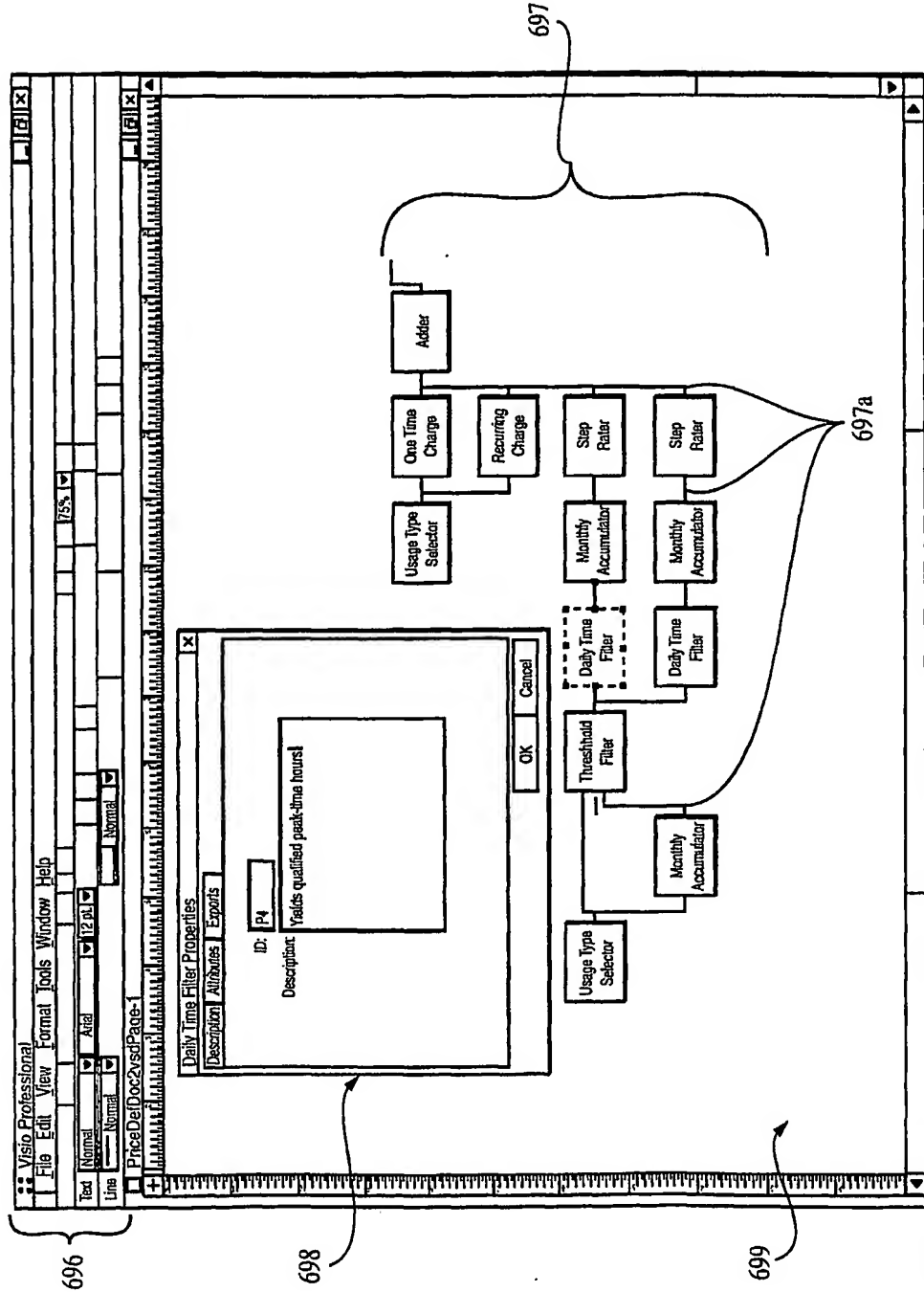


FIG. 6C

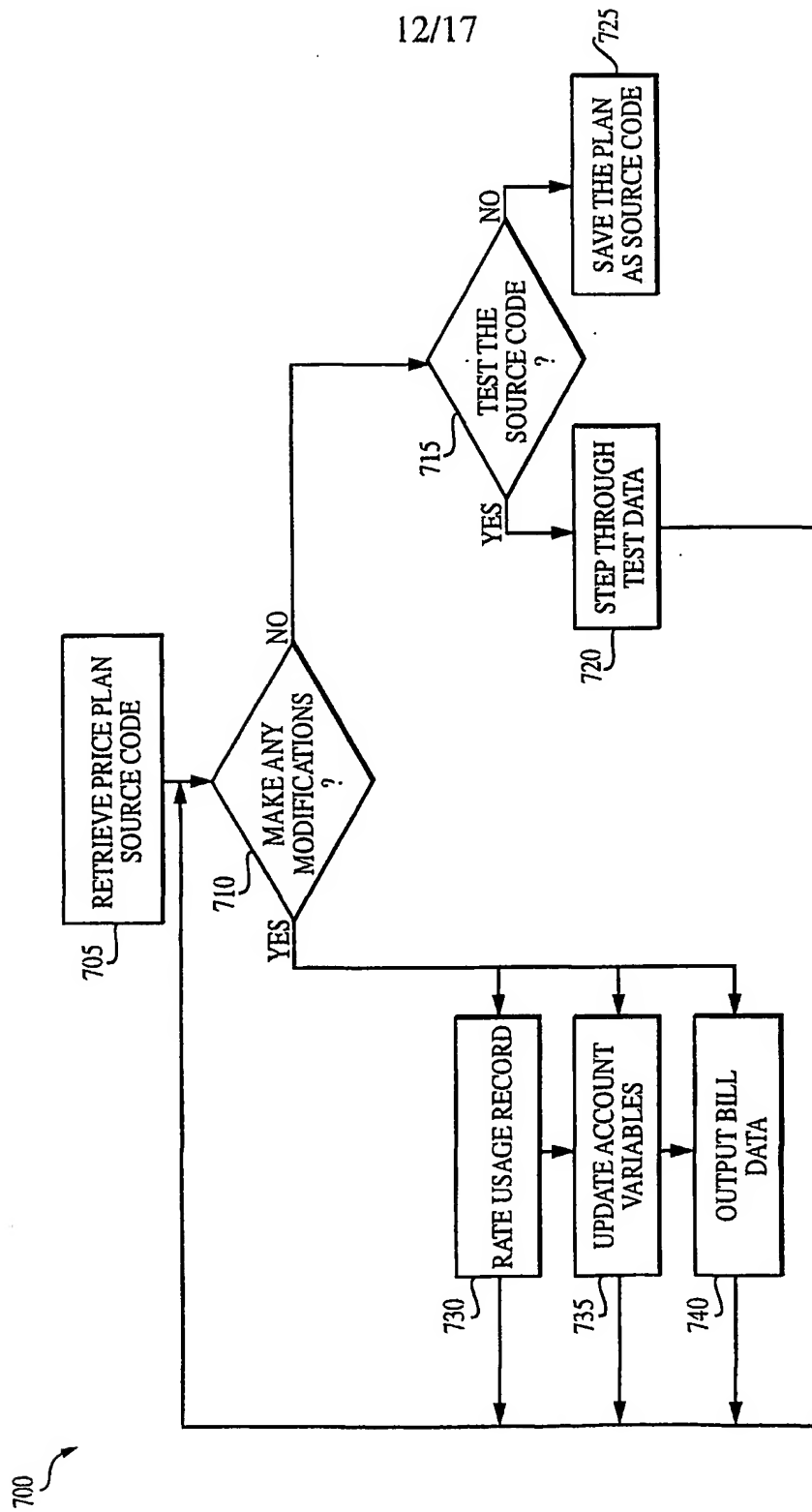


FIG. 7A

13/17

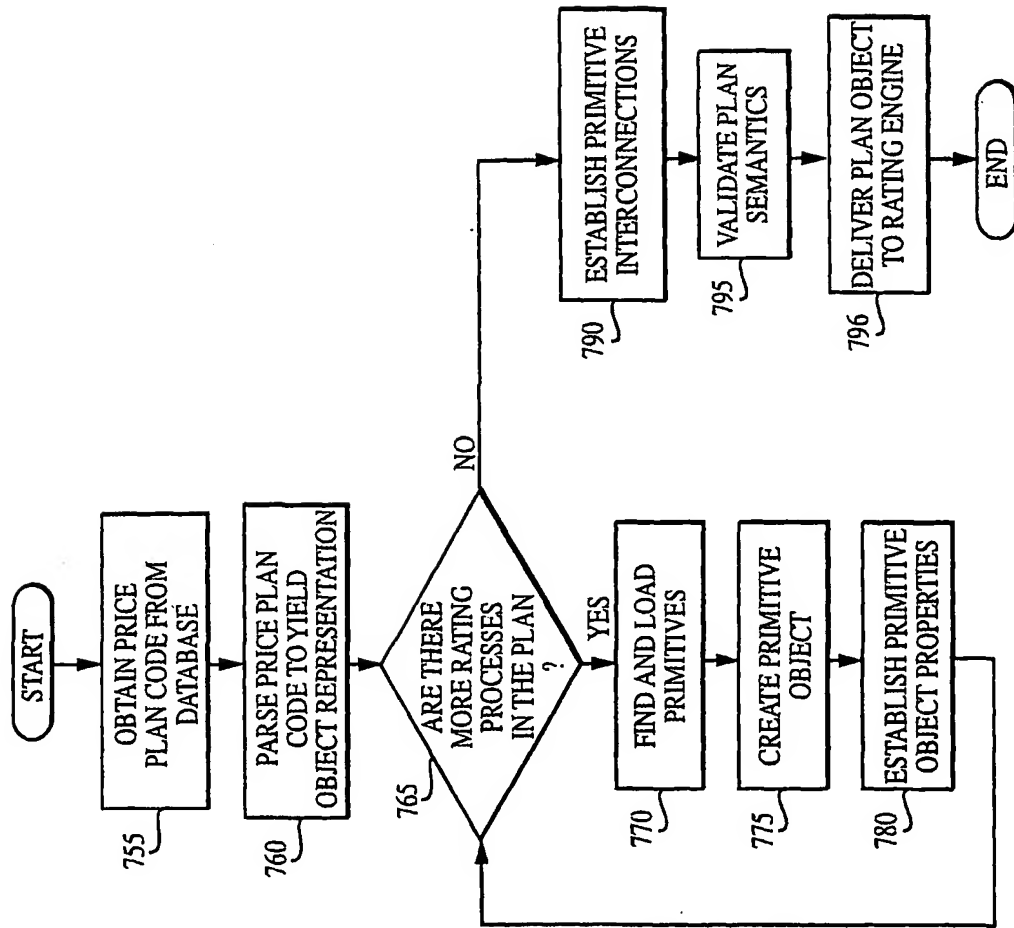


FIG. 7B

14/17

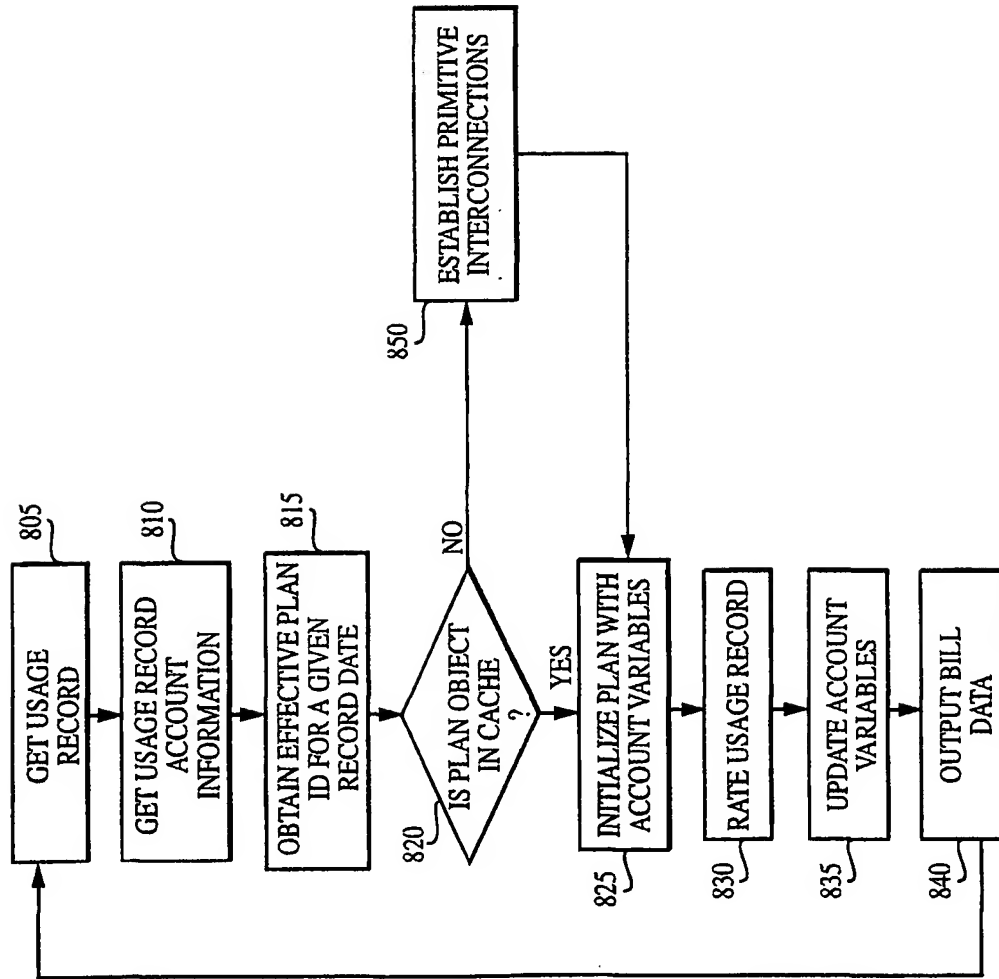


FIG. 8

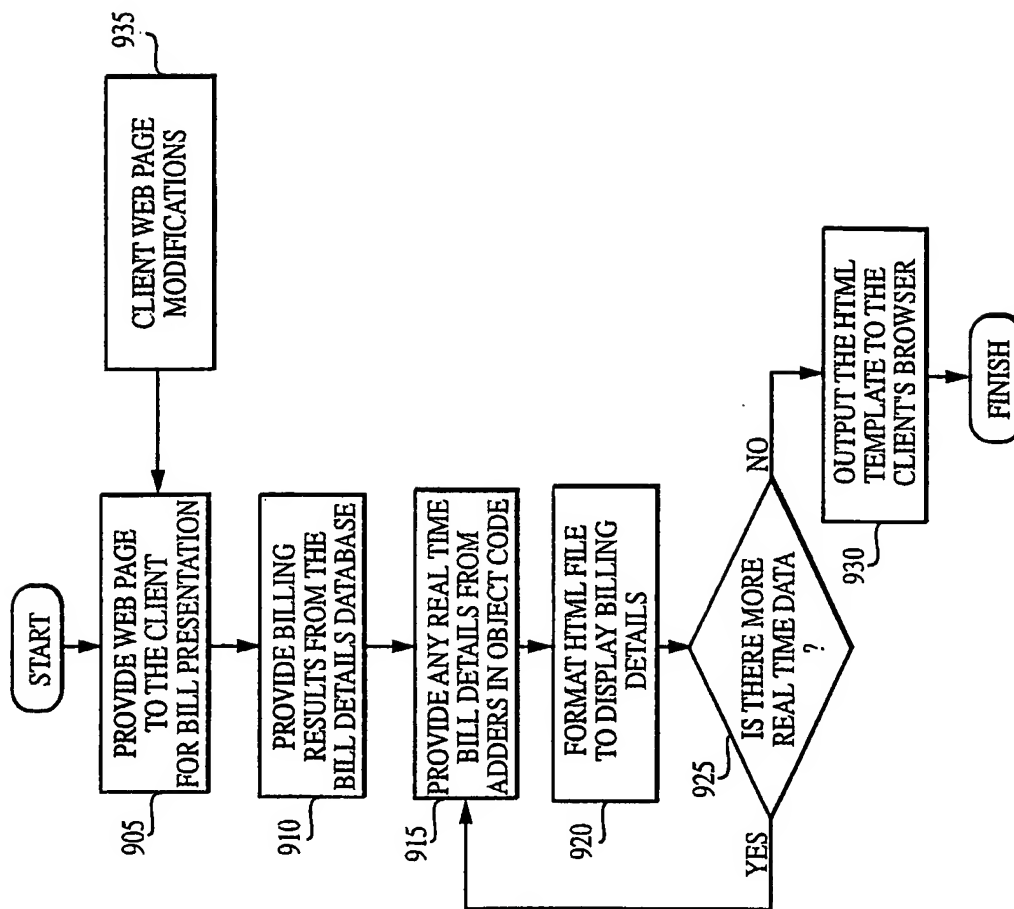


FIG. 9

16/17

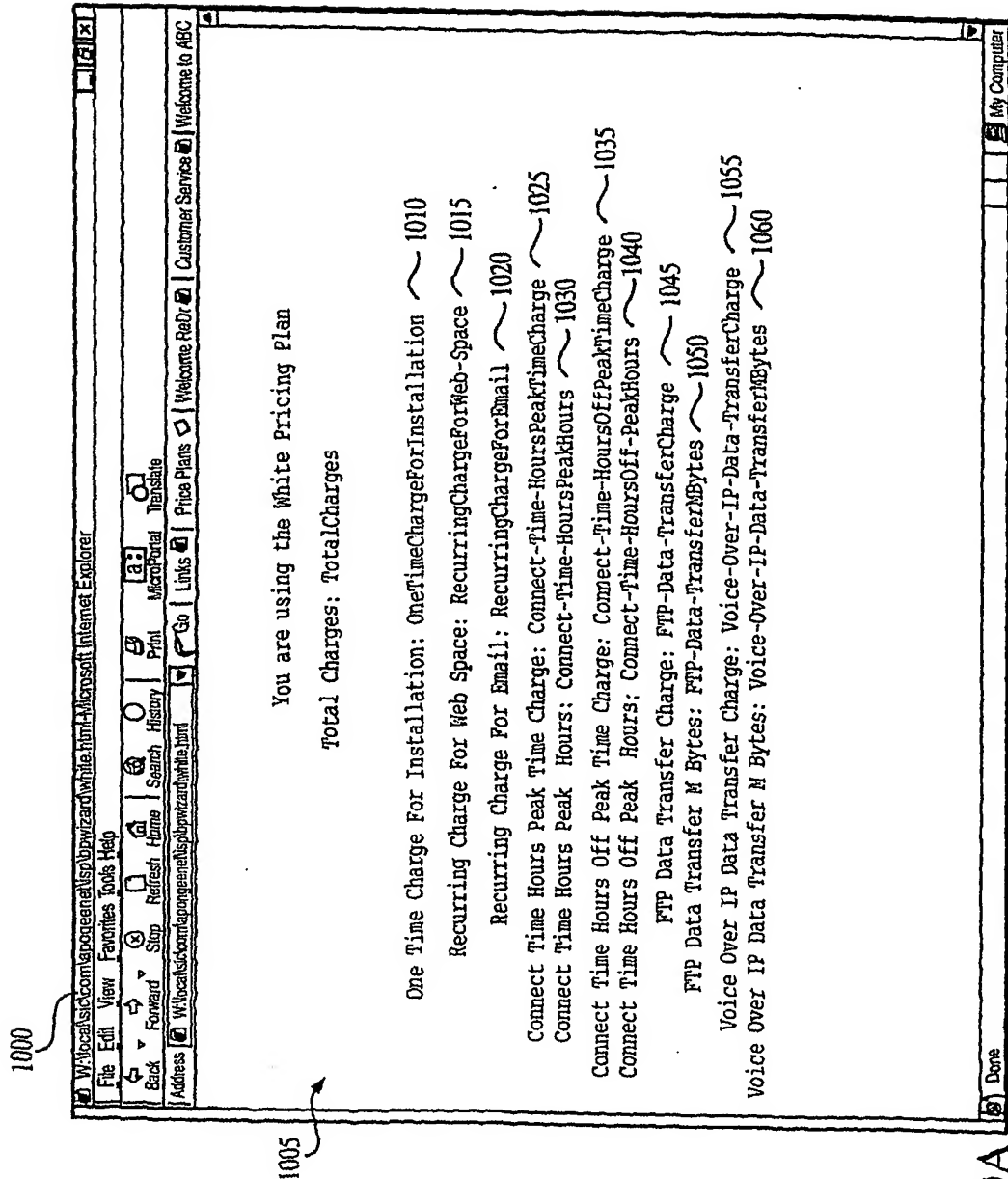
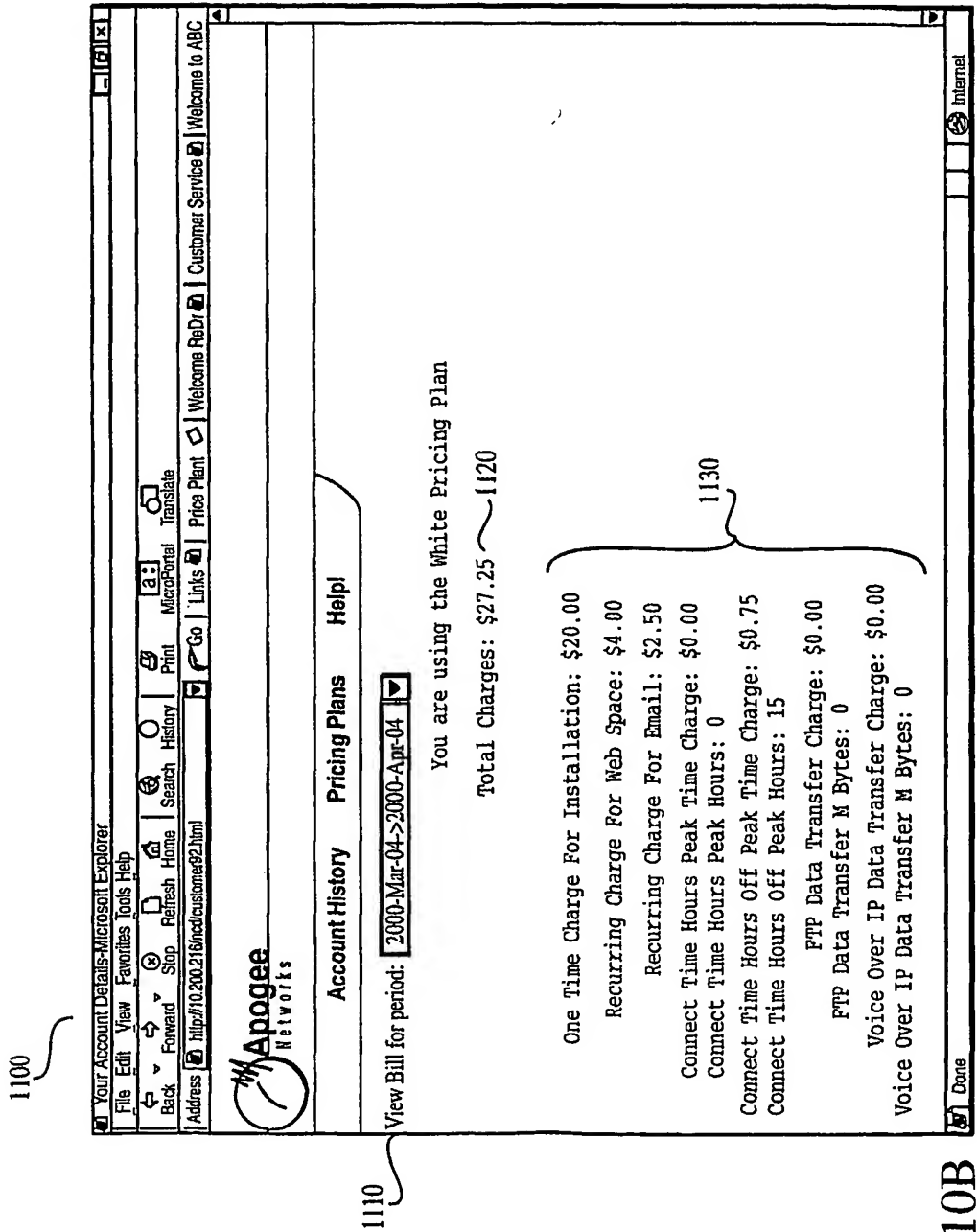


FIG. 10A

17/17



INTERNATIONAL SEARCH REPORT

 International application No.
PCT/US01/09909

A. CLASSIFICATION OF SUBJECT MATTER IPC(7) : G06F 17/60 US CL : 705/400; 717/1 According to International Patent Classification (IPC) or to both national classification and IPC		
B. FIELDS SEARCHED Minimum documentation searched (classification system followed by classification symbols) U.S. : 705/1, 400, 418; 717/1, 2, 5, 7 Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched None Electronic data base consulted during the international search (name of data base and, where practicable, search terms used) None		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	JP 05-108961 A (KOSUKEGAWA) 30 April 1993, see constitution.	1-39
A	BORJESSON: "What kind of activity-based information does your purpose require"; International Journal of Operations & Production Management, 1994, vol. 14, no. 12, pages 79-99, see abstract.	1-39
A	"NEMETSCHKE STSYEMS JOINS AEC INDUSTRY ALLIANCE FOR INTEROPERABILITY"; PR Newswire, 23 January 1996, page 123SFTU007, see lines 10-20.	1-39
A	US 5,850,221 A (MACRAE et al) 15 December 1998, see abstract.	1-39
<input checked="" type="checkbox"/> Further documents are listed in the continuation of Box C. <input type="checkbox"/> See patent family annex.		
* Special categories of cited documents: "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier document published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art "&" document member of the same patent family		
Date of the actual completion of the international search 14 AUGUST 2001		Date of mailing of the international search report 10 SEP 2001
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230		Authorized officer EDWARD R COSMANO <i>James R. Matthews</i> Telephone No. (703) 305-9783

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US01/09909

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X,P — A	US 6,205,211 B1 (THOMAS et al) 20 March 2001, see abstract.	36 ----- 1-35 & 37-39